# Rational Manager in Bitcoin Mining Pool: Dynamic Strategies to Gain Extra Rewards

Feifan Yu
Shanghai Jiao Tong University
yongyedeshuguang@sjtu.edu.cn

Na Ruan*
MoE Key Lab of Artificial Intelligence, Department of Computer Science and Engineering, Shanghai Jiao Tong University
naruan@cs.sjtu.edu.cn

Siyuan Cheng
Shanghai Jiao Tong University
516030910472@sjtu.edu.cn

## ABSTRACT

Participants of the Bitcoin system form mining pools, which have become the leading institutions of the Bitcoin mining economy, to smooth their reward of mining. Many attacks towards mining pools have been proposed. Block Withholding attack is an attacker splitting some of the power to mine in a pool, submitting shares while withholding blocks, which is one of the most famous and original attacks. Few pieces of research pay attention to the case that managers work rationally to gain extra rewards themselves at the same time of countering withholding attack. However, some different reward functions have been proposed to avoid rational miners' withholding. In this paper, we offer a model that a rational manager gain extra rewards and incentivize miners not to withhold blocks by applying a dynamic mining strategy. We conduct quantitive analysis and simulations to verify the availability and effectiveness of our attacks. We show the attack benefits the miners in the pool in some circumstances, and further discuss improvement for our attack.

## CCS CONCEPTS

• **Cryptocurrency**; • **Blockchain**;

## KEYWORDS

Pool Manager, Rational, Selfish Mining, Withholding Attack, Reward Function.

**ACM Reference Format:**
Feifan Yu, Na Ruan, and Siyuan Cheng. 2020. Rational Manager in Bitcoin Mining Pool: Dynamic Strategies to Gain Extra Rewards. In *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security (ASIA CCS '20), October 5–9, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3320269.3384754

*Corresponding Author


*ASIA CCS '20, October 5–9, 2020, Taipei, Taiwan*

ACM ISBN 978-1-4503-6750-9/20/10…$15.00
https://doi.org/10.1145/3320269.3384754

## 1 INTRODUCTION

Bitcoin, as the most representative and influential application of blockchain, is based on Proof-of-Work protocol, which takes up dominant proportions in the area of cryptocurrency. Miners use their hash power to find a number whose hash has a prefix of enough zeros. Since the total hash power of the whole network has become too large, the probability for an independent miner to find a block is tiny. Independent miners join different mining pools. Mining pools spread the risk, for their participants, smoothing their rewards, without alerting the expectation of payoff per second of computation. What miners submit to the pool manager, which can be called shares, are solutions to the new puzzle the pool defines, and the reward allocation when the pool finds a block is based on how many shares each miner has found [1]. In this way, an independent miner's reward becomes much more smooth and less variant. And the fee rate the manager takes should be relatively low, and in practice, they are usually ranged from 1 to 5 percent. Mining pools have become the dominant participants in the bitcoin system, and many pieces of research aim at the stability of mining pools.

Several kinds of attacks towards bitcoin mining protocol have been proposed. Most obviously, if a mining entity possesses more than 50% of the mining power of the whole net, it can choose any block in the main branch to begin a fork, chasing after and surpassing the main branch at last. Thus, it can double-spend a transaction no matter how long the transaction has finished without other limits. This attack is called "51% attack", which violates the spirit of decentralizing, thus avoided by participants of bitcoin economic. Another two of the most classic and original attacks are Selfish Mining attack, which attacks the whole system, and Block Withholding attack, which aims at mining pools.

Selfish Mining attack is a mining entity doesn't publish the block found immediately, but continues mining on the block and doesn't popagate it until any others find another block, when the attacker publishes private blocks, intentionally generating a fork [11]. A mining entity with more than one-third of the total mining power in the network always benefits from selfish mining, and the threshold of profiting goes down to one fourth if 50% of the power among that out of the pool follows its branch [11]. It must be emphasized that the profit of selfish mining is increasing the relative fraction of the attacker's reward in the whole network, not the reward expectation per second. As Bitcoin adjusts mining difficulties every two weeks to stabilize the generating rate to 10 minutes a block, increasing the fraction also increases the reward in a long time [13]. Different reactions to other mining entities' publishing blocks lead

to different derivative varieties of selfish mining, such as stubborn mining proposed by K Nayak et al. [13]. They improve the effect of selfish mining.

Block Withholding (BWH) attack is a mining entity can allocate part of its hashpower to participant in the target pool and submit shares except valid blocks to the manager, and use the remain hash power to mine for itself. It is proved that the attacker can always choose a fraction of allocation to get more rewards than honest mining in terms of the length of the blockchain, when the reward function of the mining pool is in proportion to the number of shares miners submit [6]. Several attacks have been proposed based on BWH attack to improve the effectiveness, such as Fork-after-Withholding (FAW) attack [9] and Power-Adjusting Withholding (PAW) attack [8]. FAW attack, combining withholding attack and selfish mining, is proved to be more profitable than selfish mining and withholding attack [9]. A countermeasure for BWH attack and some of its extension is adjusting the reward allocation function of the pool, and Okke Schrijvers et al. proposes an instance [7].

Miners' and managers' reward expectation of whether to withhold a block or selfish mine can be evaluated in long-time and short-time. Short-time-reward evaluation comes from Okke Schrijvers et al.'s pointing out that when withholding a block, a miner gives away the reward by submitting shares on mining next block during the withholding time. Short-time-reward-oriented miners care about reward expectation of a short and fixed time interval, while miners who care more about long-term rewards weight reward expectation of several blocks on the blockchain or several shares more. Most of the related works until now focus on long-term rewards. The manager being rational, attempting to gain extra rewards, is also a perspective to which few pieces of research pay attention. We study the manager's strategy in a pool consisted of short-term-reward-oriented and long-term-reward-oriented miners.

In conclusion, our contributions are:

(1) **Proposing a new model about a rational pool manager's attacking.** The manager applies a dynamic strategy to gain extra rewards, attempting to incentivize miners not to withhold blocks at the same time. More realistic, reasonable, and comprehensive assumptions are given about miners' decision on whether to withhold contrast to past researches.
(2) **Conducting quantitive analysis and simulations about the attack.** The analysis and simulations show that the attack is more profitable than the derivative of selfish mining it is based on and withholding attack. Moreover, in some circumstances, the attack benefits the miners in the pool, making the attack lastable and even attracting miners to join the pool, as a substitute for cooperating selfish mining without conspiring based on trust.
(3) **Extending the discussion to more complicated situations.** The method to avoid the miners to detect the attack is also discussed. Improving the attack and more practical and effective countermeasures remain open problems.

In this paper, Section 2 gives some preliminaries. Section 3 proposes the attack model. Section 4 analyses incentivizing miners. Section 5 studies the manager's strategy. Section 6 shows the simulation results. Section 7 gives some discussion. Section 8 conludes.

## 2 PRELIMINARIES

**Forks:** According to Bitcoin's protocol, an honest node is supposed to consider the first block received as the valid successive block and the head on which to mine. However, because of different network latencies [4], when a node builds and broadcasts a block, other nodes might build and transmit a different block before receiving the first block, leading to multiple heads of the successive chain, which is called unintentional forks. There also exists intentional forks, in which a node builds a block but withholds it until other nodes publish a block on the blockchain.
**Pooled Mining:** Miners form mining pools to lower the variance of the rewards they get. One way is miners submitting solutions that prove they are working on mining the block to the pool manager, and the pool manager distributes the block rewards to the miners when any of them submit a valid block. Usually, pooled mining includes a transaction that sends the block rewards to the address of the pool, so if miners propagate the block out of pooled mining themselves, they can neither get the block rewards nor the distributed reward for finding the valid block. Therefore, miners won't do so. Another way is p2pool protocol, which gets rid of the central node.
**Shares:** A mining pool defines a new puzzle for miners in it, whose difficulty is lower than the diffculty to find a valid block. Solutions to the puzzle submitted by miners are called shares. The rewards allocated to each miner depend on the number of shares they submit, and each share has the same probability of being a valid block.
**Power Adjusting:** In BWH attack, the attacker chooses a distribution vector of mining power split on infiltration mining in victim pools and keeps the distribution constant. However, the attacker can adjust the power distribution at the appropriate time to maintain a higher reward expectation, and this is called power adjusting. Shang Gao et al. combine power adjusting with FAW attack to propose PAW attack [8]. In our paper, miners in the pool are allowed to adjust power distribution at any time, which means disincentivizing their withholding blocks may be more difficult.

## 3 ATTACK MODELS AND ASSUMPTIONS

### 3.1 Attack Models

The attack model of this paper is a rational pool manager incentivizing miners not to withhold blocks on one hand, and attempt to get extra rewards itself on the other hand. Figure 1 shows the simplified attack model. The manager of the victim pool possesses hash power $\alpha_m$. Miners' withholding decrease others in the pool's rewards. We denote $R^i_{action}(\alpha, b, f)$ to be the reward expectation of the i-th miner choosing different actions, which depends on the array of hash power $\alpha$, that of the number of shares $b$ and the reward function $f$. The manager adjust the reward allocation function, so that $R^i_{submit}(\alpha, b, f) > R^i_{withhold}(\alpha, b, f)$ for most possible $b$, incentivizing rational miners in the pool not to withhold blocks.

The manager can track the propagation of blocks built by other miners out of the pool, increasing the propagating rate of the block formed by the victim pool using Sybil nodes [9]; thus the possibility of the pool's branch being chosen when generating a fork is increased to $\alpha_p + c * (1 - \alpha_p)$, in which $\alpha_p$ denotes the hash power of the pool and $c$ is a parameter related to the advantage

in network connections[11]. Furthermore, the manager is rational, who behaves honestly when $c$ is relatively low, but attempts to gain extra reward by selfish mining, withholding and discarding blocks found by other miners when $c$ is high enough that $R^i_{attack}(\alpha, c, b, f) > R^i_{honest}(\alpha, c, b, f)$, in which $R^m_{action}(\alpha, c, b, f)$ denotes the reward expectation of the manager, depending on $\alpha$,$c$,$b$,$f$.

The rational miners in the pool can be miners who care more about short-term rewards, focusing on reward expectation of a short and fixed time interval, and those caring more about long-term rewards that focus on reward expectation of the block and the successive block on the blockchain. The manager also needs to avoid the decreasing of miners' reward expectation in the pool as long as possible, in case rational miners hop out. We find that the manager's choice to attack reduces the reward expectation of miners out of the pool, but $\frac{R^i}{\alpha_i} > \frac{R^o}{1-\alpha_p}$ is possible, in which $R^i$ and $R^o$ denote the reward expectation of i-th miner in the pool and miners out of the pool respectively.


**Figure 1: The simplified attack model.**

## 3.2 Assumptions

We make the following assumptions to simplify our analysis, consistent with those of other attacks on Bitcoin mining, which are selfish mining [2],[3],[11], stubborn mining [13], Fork After Withholding Attack [8] and Withholding Attack [6].

(1) The block reward of a block is normalized to 1 BTC [6],[9]. The reward we calculate in the analysis is actually the reward expectation.
(2) The total mining power of the network is normalized to 1 [6],[9]. The mining power of any mining entity in the network is a fraction of this total, less than 50% to avoid "51% attack".
(3) Ignoring unintentional forks in the network, which is rational as the fork rates are negligible, about 0.41% recently [2]. The period of a mining entity finding a block is approximated as subjecting to an exponential distribution; thus the possibility of a miner finds a block or share first among multiple miners is the relative fraction its mining power among the miners.
(4) We research on centralized mining pools. When a pool manager propagates and publishes a block, it distributes the block rewards to the miners in the pool based on the number of shares they submit in this round.
Specifically, in our problem, we make the following assumptions also.
(5) Each miner in the target pool possesses less than 33% power of the pool, so that selfish mining is not profitable for the miner even if we don't consider miners out of the pool finding a block.
(6) Attackers who launch a withholding attack to other pools are not taken into consideration, as well as other attackers except the manager launching other attacks such as stubborn mining and fork-after-withholding. Attackers launch withholding attack to the pool and don't take other attackers' withholding into consideration.
(7) Aside from honest miners, selfish and rational miners who may launch withholding attack exist in the pool. They may care more about short-term rewards, evaluating a choice in terms of reward expectation in a short and fixed time interval. They may care more about long-term rewards, who evaluate a decision in term of reward expectation of several blocks on the blockchain or several shares.
(8) All the shares have the same possibility to be a valid block. The manager adjusts the reward allocation function to avoid miners' withholding. Most pools apply Pay-per-share allocation function in reality, and it might be more attractive for miners, but it may lead to the pool manager's loss when the time to find a block is too long, which may indicate the manager to charge a higher fee rate. The manager is assumed to choose from two functions: adjusted proportional function, which we propose an analysis later, and IC reward function proposed by Okke Schrijvers et al.
(9) Each miner in the pool includes a transaction that declares it finds the share/block when calculating the hash and submitting the share/block, in case the manager plunders the reward by declaring it to be found by the manager itself.

# 4 INCENTIVIZING MINERS

The manager's first goal is to prevent most miners' withholding blocks, which leads to other participants in the pool's loss. Therefore, the manager needs to consider miners' choices under different reward allocation functions.

## 4.1 Overview of Miner's Choice

In our model, a rational miner can distribute its mining power into infiltration mining and innocent mining. Infiltration mining works on the puzzle defined by the pool and may withhold blocks. Innocent mining means mining as an individual miner, getting all the block rewards when finding a block. A rational miner can adjust the splitting distribution of mining power, finding the splitting distribution with the highest withholding reward expectation. The reward allocation function we propose is that the manager adjusts the reward allocation function on the base of proportional allocation function: increasing the rewards for the miner who submits a valid block, to achieve the first goal. We define the parameters in Table 1 to analyze miners' choice.

For a rational miner who focuses on short-term rewards, Okke Schrijvers et al. proves a lemma based on the assumption of all mining power being in the same pool shown as follows [6]:

Lemma 4.1. *For a reward allocation function R, a player i has an incentive to report full solutions immediately, iff the following*

*condition holds for all* $\{\alpha_i\}_{i=1}^n$, $b_t$, $D$, $i$:

$$\sum_{j=1}^{n} \alpha_j \cdot (R_i(b_t + e_j) - R_i(b_t)) \leq \frac{E_b[R_i(b)]}{D},$$

*in which* $e_j$ *means* $j^th$ *standard basis vector that is 0 everywhere except for the* $j^th$ *component, and* $E_b[R_i(b)$ *means the reward expectation player i gets in the next block when anyone finds it.*

$\frac{E_b[R_i(b)]}{D}$ in lemma 4.1 is the reward expectation player $i$ gets in the expectation of time interval to find a share. In other words, lemma 4.1 tells that a rational miner who focuses on short-term rewards would submit the valid block immediately if and only if the reward of submitting immediately is higher than that of waiting for a share [6]. Though its proof is based on the assumption of all mining power is in the same pool, and the miners compute rewards in terms of time, we can extend it to under our circumstance and for rational miners who focus on long-term rewards.

We neglect the manager's managing fee to simplify the analysis, which means $q_t$ satisfies $s * \frac{\sum b_i^t + q_t}{\sum b_i^t + 1} = 1$. The manager should choose appropriate $s$ so that the rewards of submitting the block immediately is larger than that of waiting for a share. The smaller $s$ is, the less the relative profit the miner can get from withholding the block and continuing submitting shares is. Thus, the appropriate $s$ is smaller than a threshold value. On the other hand, the miners join mining pools to smooth their rewards, while small $s$ leads to a big difference in rewards between whether finding a valid block, distracting them. Therefore, the manager's central work in this step is to find the appropriate $s$, smaller than a threshold but not too small.

Intuitively, miners with little hash power are more likely to submit the block immediately, because it's difficult for them to find the next block. In this thought, the manager only needs to find $s$ that the largest miner in the pool would submit at once. Another intuitive verdict is that the smaller the number of shares the target miner has found is, the more likely he is to withholding the block, expecting to increase the fraction of shares among all.

**Table 1: The parameters defined**

| | |
|---|---|
| $\alpha_p$ | The total nominal power of the pool, including a miner in the pool's power of infiltration mining and innocent mining, larger than power of any miner in the pool |
| $\alpha_i$ | The hashpower of the i-th miner in the pool. |
| $D$ | The expectation of time to find a block, devided by that of finding a valid share. |
| $s$ | Adjustment parameter to the proportional reward allocation funtion which rewards for shares are multiplied by, the rest of rewards distributed to the valid block founder admitted by the manager |
| $b_t$ | The array of number of shares miners have submit in this block's mining at time t. |
| $b_i^t$ | The i-th miner's number of shares in this block's mining at time t. |
| $r$ | The fraction of a miner's mining power allocated in the pool in all power owned. |
| $q_t$ | Adjustment to the proportional reward allocation funtion: rewards for a valid block devided by rewards for a share. |

## 4.2 Incentivizing Short-Term-Reward-Oriented Miner

This subsection analyses short-term-reward-oriented miners' decisions. Since short-term-reward-oriented miners' choices under IC reward function has been analyzed by Okke Schrijvers et al.[6], the analysis focuses on adjusted proportional function. We denote $T$ to be the time expectation for the pool to find a share when miner $i$ allocates some of the power on innocent mining. A Short-term-reward-oriented miner identified miner $i$'s expectation of rewards in $T$ of submitting the block immediately is approximate to:

$$R_{submit}^i = s * \frac{b_i^1 + q_1}{\sum b_j^1 + 1} + \frac{\alpha_i}{D * (\alpha_p - (1-r) * \alpha_i)} \tag{1}$$

Proof. The miner can be allocated $s * \frac{b_i^1 + q_1}{\sum b_j^1 + 1}$ in this block if submitting the block immediately. The time of the pool to find a share, labled $T$, when miner $i$ allocates some of the power on innocent mining subjects to an exponential distribution, which is approximate to the linear distribution when $T$ is near 0. Thus, when D is large, the reward expectation in $t$ to find a share when miner $i$ allocates some of the power on innocent mining is approximate to $\frac{\alpha_m}{\alpha_p - (1-r)*\alpha_i}$ of reward expectation of a share, which is $\frac{1}{D}$. And the expectation of rewards in $T$ is the sum of the above two parts. □

We denote $P_1$ to be the probability of no other miners out of the pool find block in $T$, which means $P_1 = [1 - \frac{1}{D}]^{\frac{1-\alpha_p+(1-r)*\alpha_i}{\alpha_p-(1-r)*\alpha_i}}$, and $P_2$ to be the possibility of the miner finding next share, which means $P_2 = \frac{r*\alpha_i}{\alpha_p-(1-r)*\alpha_i}$. The miner's reward expectation of waiting for one share is:

$$\begin{aligned} R_{wait}^i = P_1 * [(1 - \frac{1}{D}) * s * \frac{(b_i^1 + q_2 + P_2)}{\sum b_j^1 + 2} \\ + \frac{1}{D} * [P_2 * (s * \frac{b_i^1 + q_1}{\sum b_j^1 + 1}) + (1 - P_2) * s * \frac{b_i^1}{\sum b_j^1 + 2}]] \\ + (1 - P_1) * \frac{(1-r) * \alpha_i}{1 - \alpha_p + (1-r) * \alpha_i} \end{aligned} \tag{2}$$

Proof. Formula (2): Assume the miners out of the pool also mine for shares, and approximate the time distribution for a mining entity to find a share to be linear distribution. The number expectation of shares found by miners out of the pool is approximated to be $\frac{1-\alpha_p+(1-r)*\alpha_i}{\alpha_p-(1-r)*\alpha_i}$, so $P_1 = [1 - \frac{1}{D}]^{\frac{1-\alpha_p+(1-r)*\alpha_i}{\alpha_p-(1-r)*\alpha_i}}$. When miners out of the pool, including miner $i$'s innocent mining power, don't find a block during the time: if the share found by the pool is not a valid block, which has a probability of $(1 - \frac{1}{D})$, the reward expectation is $\frac{s*(b_i^1+q_2+P_2)}{\sum b_j^1+2}$; if it is a valid block, which has a probability of $\frac{1}{D}$, the

reward expectation is $P_2 * (s * \frac{b_i^1+q_1}{\sum b_j^1+1}) + (1-P_2) * s * \frac{b_i^1}{\sum b_j^1+2}$. When miners out of the pool find a block during the time: the reward expectation is $\frac{(1-r)*\alpha_i}{1-\alpha_p+(1-r)*\alpha_i}$. Thus Formula (2) is derivated. □

When the reward expectation of submitting the block is higher, the incentivizing is successful.

### 4.3 Incentivizing Long-Term-Reward-Oriented Miner

*4.3.1 Adjusting Proportional Reward Function.* This subsection analyses long-term-reward-oriented miners' decisions. This type of miners are more likely to withhold the block they find, and they don't take the expectation of reward they would get from mining on the next block during the withholding time if they publish the block immediately into consideration seriously. Thus, the opportunity cost of withholding for them is lower.

The results are different from Loi Luu et al.'s [4] because we allow the attacker to adjust power splitting at any time; thus, it's more challenging to achieve that goal. A miner of this type's expectation of rewards on the following two blocks on the blockchain of submitting the block immediately is:

$$R^i_{submit} = s * \frac{b_i^1 + 1 + q_1}{\sum b_j^1 + 1} + \alpha_i \quad (3)$$

$P_1$ and $P_2$ are the same to those in Section 4.2, and the reward expectation on the following two blocks on the blockchain of waiting for a share is:

$$R^i_{wait} = P_1 * [(1-\frac{1}{D}) * [\alpha_i + s * (b_i^1 + q_2 + P_2)/(\sum b_j^1 + 2)] + \frac{1}{D} * [P_2 * (\alpha_i + s * \frac{b_i^1+q_1}{\sum b_j^1+1}) + (1-P_2) * (\alpha_i + \frac{s * b_i^1}{\sum b_j^1+2})]] + (1-P_1) * [\frac{(1-r)*\alpha_i}{1-\alpha_p+(1-r)*\alpha_i} + \alpha_i] \quad (4)$$

When the reward expectation of submitting the block is higher, the goal is achieved. The derivation process of Formula (3),(4) is similar to that of Formula (1),(2). The only significant difference is that miner $i$'s reward expectation in the next block is $\alpha_i$.

*4.3.2 IC Reward Function.* A long-term-reward-oriented miner's expectation of rewards on the following two blocks on the blockchain of submitting the block immediately is:

$$R^i_{submit(ic)} = \frac{b_i + 1 - (\sum b_i^1 + 1)}{max(D, \sum b_i^1 + 1)} + 1 + \alpha_i \quad (5)$$

$P_1$ and $P_2$ are the same to those in Section 4.2, and the reward expectation of waiting for a share is:

$$R^i_{wait} = P_1 * [(1-\frac{1}{D}) * [\alpha_i + 1 + \frac{b_i + 1 + P_2 - (\sum b_j^1 + 2)}{max(D, \sum b_j^1 + 2)}] + \frac{1}{D} * [P_2 * (\alpha_i + 1 + \frac{b_i^1 + 1 - (\sum b_j^1 + 1)}{maxD, \sum b_j^1 + 1}) + (1-P_2) * (\alpha_i + \frac{b_i^1}{max(D, \sum b_j^1 + 2)}] + (1-P_1) * [\frac{(1-r)*\alpha_i}{1-\alpha_p+(1-r)*\alpha_i} + \alpha_i] \quad (6)$$

The derivation process of Formula (5),(6) is similar to that of Formula(3),(4), replacing the allocated reward in this block with that allocated by IC reward function. The manager's next step is to adjust mining strategies dynamically to gain extra reward on the base of almost no miners in the pool withhold blocks, which we analyze in Section 5.

## 5 MANAGER'S STRATEGY

### 5.1 Overview of Manager's Choice

The manager behaves the way honest managers do when another pool publishes a block and the manager doesn't have withheld block in hand. But when the manager finds a block ahead of other miners in and out of the pool, it can split the mining power into three choices if not publishing it at once:

Choice A: **withholding the block, and continuing submitting the shares while splitting some of the hash power owned on individual innocent mining**, in other words, conducting a withholding attack itself. As the manager can disable blocks submitted by other miners in its pool during the withholding and has knowledge about the advantage it has in network connections, the reward expectation of withholding is different from that of miners.

Choice B: **selfish mining on the block found, and still setting the address to receive rewards to be the pool address until some miners find a block.** If miners not in the pool find the block, the manager publishes the block as soon as possible to make a fork. If the manager finds the second block, it publishes two blocks together. If other miners in the pool find the block, the manager can either publish the first block, or discard the block submitted by the miner, and continue selfish mining until miners out of the pool find a block.

Choice C: **selfish mining on the block found, setting the address to receive rewards in the second block to be its own address, and then the same as Choice B.**

Whether the reward allocation function in our model is adjusted proportional reward function or IC reward function, since no other miner submits shares in the second block, the rewards the manager and others gain in Choice C is the same as that in Choice B. So only the rewards of Choice A and one of Choice B and C are needed to be compared in our model.

When the other miners in the pool find and submit a block first, the manager sees it as a block on the private chain of the pool, facing the same choices as above.

To simplize the model, selfish mining with part of the power and withholding with other power at the same time is not considered in this section. And the reward expectation of it can be computed with that of selfish mining and withholding. We define the following additional parameters in Table 2 to analyze the manager's choice.

Managers who care about short-time rewards and long-time-reward-oriented managers' strategies are analyzed in Section 5.2 and 5.3, respectively.

### 5.2 Manager cares about short-time rewards

If the manager cares more about short-time rewards, it would calculate the expectation of rewards in the term of a fixed time.

Comparing the reward expectation in different cases between Choice A and Choice B, we can easily find that:

**Table 2: The additional parameters defined**

| | |
|---|---|
| $c$ | The fraction of mining power who would follow the miner's branch when there exists a fork among the whole network out of the pool. |
| $\alpha_m$ | The fraction of the manager's mining power in the whole network. |
| $b_0^t$ | The number of the manager's shares in this block's mining at time t. |
| $r_m$ | The fraction of the manager's mining power allocated in the pooled mining. |

Theorem 5.1. *If anyone in the pool finds a new block, a manager caring about short-time rewards won't split any of the power on individual mining the block of the same height.*

Proof. The possibility and time distribution of the manager's innocent mining power finding another block ahead of miners out of the pool are the same as those of allocating the power to mine the next block of the private block withheld. However, the manager can get only a block's reward if its innocent mining power succeeds to find the next block of the original block on the public blockchain while it can get a block's reward and some allocated reward for shares if succeeding in the latter choice. If miners out of the pool publish a new block first, the loss of both choices is the same. And if other miners in the pool submit a new block first, the manager would discard it in both choices. □

So the manager only needs to choose from honest behavior and selfish mining.

When the manager finds a block before other miners in the network, if the reward function is adjusted proportional reward function, the reward expectation of these choices in the time for the rest of power in the pool to find a share is shown as follows:

(1) publishing immediately:

$$R_{pub}^{ms} = s * \frac{b_0 + q_3}{\sum b_i + 1} + \frac{\alpha_m}{D * (\alpha_p - \alpha_m)} \tag{7}$$

The derivation process of Formula (7) is similar to that of Formula (1).

(2) selfish mining: we denote $P_3$ to be the possibility of no miners out of the pool find a block during the time, which means $P_3 = [1 - \frac{1}{D}]^{\frac{1-\alpha_p}{\alpha_p - \alpha_m}}$; and $R_3$ to be the reward the manager would gain in this block if publishing it immediately, which means $R_3 = s * \frac{b_0+q_3}{\sum b_j^1+1}$. The reward expecation of selfish mining is:

$$R_{sm}^{ms} = P_3 * [(1 - \frac{1}{D}) * s * \frac{b_0 + q_2}{\sum b_j^1 + 2} + \frac{1}{D} * (\frac{\alpha_m}{\alpha_p} + R_3)] + (1 - P_3) * (\alpha_p + c * (1 - \alpha_p)) * R_3 \tag{8}$$

Proof. According to Theorem 5.1, the miner would allocate all of the power to mine on the private block if choosing on selfish mining, so $P_3 = [1 - \frac{1}{D}]^{\frac{1-\alpha_p}{\alpha_p - \alpha_m}}$. If the pool finds another block in the time interval: when the block is found by the manager, the reward expectation is $\frac{\alpha_m}{\alpha_p} + R_3$; when the block is found by others, the reward expectation is $R_3$. □

If the reward function is IC reward function, the reward expectation of these choices in the time for the rest of power in the pool to find a share is shown as follows:

(1) publishing immediately:

$$R_{pub(IC)}^{ms} = \frac{b_0 + max(D, \sum(b_i) + 1) - (\sum(b_i) + 1)}{max(D, \sum(b_i) + 1)} + \frac{\alpha_m}{D * (\alpha_p - \alpha_m)} \tag{9}$$

(2) selfish mining: we denote $P_3$ to be the same as above; and $R_{3(IC)}$ to be the reward the manager would gain in this block if publishing it immediately, which means $R_{3(IC)} = \frac{b_0+max(D,\sum(b_i)+1)-(\sum(b_i)+1)}{max(D,\sum(b_i)+1)}$. The reward expecation of selfish mining is:

$$R_{sm(IC)}^{ms} = P_3 * [(1 - \frac{1}{D}) * \frac{b_0 + max(D, \sum(b_i) + 2) - (\sum(b_i) + 2)}{max(D, \sum(b_i) + 2)} + \frac{1}{D} * (\frac{\alpha_m}{\alpha_p} + R_{3(IC)})] + (1 - P_3) * (\alpha_p + c * (1 - \alpha_p)) * R_{3(IC)} \tag{10}$$

The manager would choose the choice with higher reward expectation. The case when other miners in the pool find a block first is similar, and that is left out in consideration of the length of the paper. Comparing the reward formulas, We find that this kind of manager would never conduct a selfish mining attack.

Theorem 5.2. *Short-term-reward-oriented manager will not do selfish mining attack, regardless of the mining power distribution and the number of shares.*

Proof.

$$R_{sm}^{ms} \leqslant P_3 * [(1 - \frac{1}{D}) * s * \frac{b_0 + q_2}{\sum b_j^1 + 2} + \frac{1}{D} * (\frac{\alpha_m}{\alpha_p} + R_3)] + (1 - P_3) * R_3$$

$$\leqslant P_3 * [(1 - \frac{1}{D}) * R_3 + \frac{1}{D} * (\frac{\alpha_m}{\alpha_p} + R_3)] + (1 - P_3) * R_3$$

$$= R_3 + \frac{1}{D} * \frac{\alpha_m}{\alpha_p} \leqslant R_3 + \frac{\alpha_m}{D * (\alpha_p - \alpha_m)} = R_{submit}^{ms}$$

□

An explanation for this is the action of selfish mining wastes the network's mining power; thus, it's hard to increase short-term rewards in a fixed time interval. And the mining power of the manager is less than standard selfish mining requires. Selfish mining always decreases its rewards in terms of time.

## 5.3 Manager cares about long-time rewards

If the manager cares more about long-time rewards, it tends to calculate the expectation of rewards in the term of blockchain length.

The expectation of reward of withholding is different from that in section 4.3 because the manager knows c and can neglect or discard blocks found by other miners in the pool when the manager withholds and submits shares. If the reward function is adjusted proportional reward function, when the manager finds a block before other miners in the network, the expectation of reward in this

and the successive block of these two choices is shown as follows:

(1) publishing immediately:

$$R^{ml}_{pub1} = s * \frac{b_0 + q_3}{\sum b_i + 1} + \alpha_m \tag{11}$$

(2) selfish mining until others finds a block: we denote $sum2$ to be the expecation of the total number of shares all miners submit in the pool, and $q_4$ to be the corresponding parameter of reward for a valid FPoW, and $E_1$ to be the expectation of $s * \frac{b_0+q_4}{sum2+1}$. The reward expection is:

$$R^{ml}_{sm1} = (1-\alpha_p) * [\alpha_p * (E_1 + \frac{\alpha_m}{\alpha_p}) + c * (1-\alpha_p) * E_1] + \alpha_m * [1 + E_1] + (\alpha_p - \alpha_m) * (\alpha_m + E_1) \tag{12}$$

If the reward function is IC reward function, the expectation of reward in this and the successive block of these two choices is shown as follows:

(1) publishing immediately:

$$R^{ml}_{pub1(IC)} = \frac{b_0 + max(D, \sum(b_i)+1) - (\sum(b_i)+1)}{max(D, \sum(b_i)+1)} + \alpha_m \tag{13}$$

(2) selfish mining until others finds a block: we denote $sum2$ to be the same as above, and $E_{1(IC)}$ to be the expectation of $\frac{b_0+max(D,sum2+1)-(sum2+1)}{max(D,sum2+1)}$. The reward expection is:

$$R^{ml}_{sm1(IC)} = (1-\alpha_p) * [\alpha_p * (E_{1(IC)} + \frac{\alpha_m}{\alpha_p}) + c * (1-\alpha_p) * E_{1(IC)}] + \alpha_m * [1 + E_{1(IC)}] + (\alpha_p - \alpha_m) * (\alpha_m + E_{1(IC)}) \tag{14}$$

The manager would choose the choice with a higher reward expectation. The derivation process of Formula (11),(13) is similar to that of Formula (1),(2),(3),(4). The only two significant differences are that the manager would publish the block withheld if other miners in the pool find another block first, becoming the founder of the block, and if miners out of the pool find another block first, the branch of whose wins in the fork with the probability of $\alpha_p + c * (1-\alpha_p)$. The derivation process of Formula (12),(14) is similar to that of Formula (8).

The case when other miners in the pool find a block first is similar. It should be emphasized that the reward expectation of selfish mining computed in this way can't be compared to that of withholding directly, as withholding until a different number of shares found may lead to a different reward.

If the reward function is adjusted proportional function, we denote $P_3$ and $R_3$ the same to those in Section 5.2, and $P_5$ to be the possibility of no miners out of the pool finds another block during the period the pool finds a share if the manager chooses withholding, which means $P_5 = [1 - \frac{1}{D}]^{\frac{1-\alpha_p+(1-r_m)*\alpha_m}{\alpha_p-(1-r_m)*\alpha_m}}$. The reward expectation of the three choices until next share found in the following two blocks on the blockchain is as follows:

(1) publishing immediately:

$$R^{ml}_{pub2} = R_3 + \alpha_m \tag{15}$$

(2) selfish mining:

$$R^{ml}_{sm2} = P_3 * [(1 - \frac{1}{D}) * [s * \frac{b_0 + q_2}{\sum b^1_j + 2} + \alpha_m] + \frac{1}{D} * [R_3 + \frac{\alpha_m}{\alpha_p} + (1 - \frac{\alpha_m}{\alpha_p}) * \alpha_m]] + (1-P_3) * [\alpha_p * (\frac{\alpha_m}{\alpha_p} + R_3) + c * (1-\alpha_p) * R_3] \tag{16}$$

(3) withholding:

$$R^{ml}_{hold2} = P_5 * [(1 - \frac{1}{D}) * (s * \frac{b_0 + q_2 + \frac{r_m * \alpha_m}{\alpha_p - (1-r_m)*\alpha_m}}{\sum b^1_j + 2} + \alpha_m) + \frac{1}{D} * (\alpha_m + R_3)] + (1-P_5) * [\frac{1-\alpha_p}{1-\alpha_p+(1-r_m)*\alpha_m} * [\alpha_p * (\frac{\alpha_m}{\alpha_p} + R_3) + c * (1-\alpha_p) * R_3] + [\frac{(1-r_m)*\alpha_m}{1-\alpha_p+(1-r_m)*\alpha_m} * (1+\alpha_m)]] \tag{17}$$

If the reward function is IC reward function, the reward expectiation of the three choices is as follows:

(1) publishing immediately:

$$R^{ml}_{pub2} = R_3 + \alpha_m \tag{18}$$

(2) selfish mining:

$$R^{ml}_{sm2} = P_3 * [(1 - \frac{1}{D}) * [s * \frac{b_0 + q_2}{\sum b^1_j + 2} + \alpha_m] + \frac{1}{D} * [R_3 + \frac{\alpha_m}{\alpha_p} + (1 - \frac{\alpha_m}{\alpha_p}) * \alpha_m]] + (1-P_3) * [\alpha_p * (\frac{\alpha_m}{\alpha_p} + R_3) + c * (1-\alpha_p) * R_3] \tag{19}$$

(3) withholding:

$$R^{ml}_{hold2} = P_5 * [(1 - \frac{1}{D}) * (s * \frac{b_0 + q_2 + \frac{r_m * \alpha_m}{\alpha_p - (1-r_m)*\alpha_m}}{\sum b^1_j + 2} + \alpha_m) + \frac{1}{D} * (\alpha_m + R_3)] + (1-P_5) * [\frac{1-\alpha_p}{1-\alpha_p+(1-r_m)*\alpha_m} * [\alpha_p * (\frac{\alpha_m}{\alpha_p} + R_3) + c * (1-\alpha_p) * R_3] + [\frac{(1-r_m)*\alpha_m}{1-\alpha_p+(1-r_m)*\alpha_m} * (1+\alpha_m)] \tag{20}$$

It should be emphasized that the value of Formula (17) and (20) lower than those of the other two in the same group doesn't mean selfish mining is not profitable, because selfish mining for a longer time if miners out of pool haven't found blocks may lead to higher reward. The derivation process of the formulas above is similar to that of Formula (1)-(8).

What's more, the manager can apply this strategy of whether to withhold and discard other miners' blocks in more advanced and comprehensive attacks. Take stubborn mining as an instance. When any miners in the pool find a block, the manager can compute reward expectations and decide whether to withhold it. The possibility of the pool finding a block before miners out of the pool is larger than the manager mining alone, and catching up when falling on the behind can be easier as miners in the pool who would get relatively high rewards if the branch of the pool becomes the main branch. Whether it's more beneficial than stubborn mining alone for the manager depends on the hash power of the pool as well as

the manager and c, as the miners may turn to innocent mining on the public branch when the pool falls behind, if they think catching up is difficult. And it also depends on the reward allocation function parameters of the pool. Deeper analysis is left out in this paper, but the manager may get more extra rewards applying this comparing to delivering those attacks alone in a suitable environment, especially when the manager can charge administrating fees.

## 6 SIMULATION AND ANALYSIS

We evaluate the availability and effect of the attack from the perspective of the manager by simulation. Section 6.1 shows the effect of adjusting the parameters on incentivizing miners not to withhold blocks. We implement a Monte Carlo simulator in Python and run the simulator over $10^8$ rounds and display the results in Section 6.2 and 6.3. The hash power of the pool is changed from 0.3 to 0.4 in our simulations, and the nodes out of the pool are set to be honest to decrease the difficulty. The number of variables makes it difficult to set the simulation clear, increasing the difficulty to evaluate the practicality of the proposed attack. We try to give an explanation about what it needs to launch an attack, and how to define whether the other conditions are consistent, comparing the effect with that of attacking alone.

### 6.1 Quantitive Analysis of Incentivize Miners

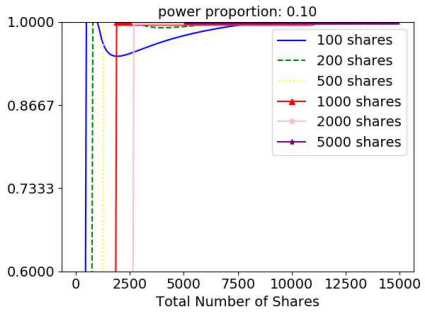
We denote $t_0$ to be the time any miner in the network finds a block, $sum_3^t$ to be $\sum b_i^t$, and $\alpha_{-i}$ to be $(\alpha_p - (1-r) * \alpha_i)$. $sum_3^{t_0}$ can be seen as a random variable, and it is approximated that $sum_3^{t_0} \sim Ge(\frac{1}{D})$.

It is also approximated $b_t \sim PN(sum_3^t, \frac{\alpha_m}{\alpha_{-i}}, \frac{\alpha_1}{\alpha_{-i}}, \cdots, \frac{\alpha_n}{\alpha_{-i}})$ when miner $i$ is withholding blocks. The approximations are reasonable as each share has the same possibility of $\frac{1}{D}$ to be a valid block, and each miner's number of shares is proportional to their hash power. The approximations also satisfy the limitation that the sum of the number of shares by miners is the total number of shares.

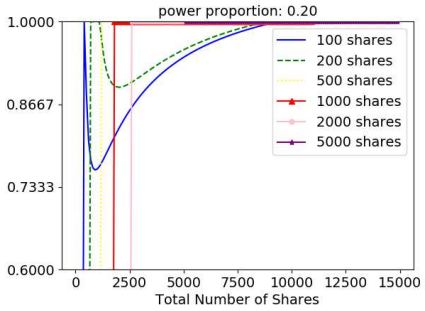
Approximately regarding the number of shares as other distributions is also considered, but it is found that it may lead to problems. For instance, if applying Poisson distribution, the time for the whole network to find a new block should also be approximated as relatively steady and should be shorter than that of the pool; thus the pool always stops mining and move to the next block as other mining power finds a block first. And it is a paradox. If approximating the time for the pool to find a share is relatively steady, and the numbers of shares different miner and the whole pool have found subject to different exponential distributions whose expectation are proportional to the miner's mining power in the pool, the accumulative reward comes out to be not proportional to mining power even when the simple proportional reward function is applied. Thus, the approximation adapted now is better than these two.

It should be noted that the whole pool can be seen as a miner with the mining power of the whole pool; thus, the process of miner and the pool finding shares should be of the same distribution. However, the termination time of counting is when the whole pool finds a valid block, which one miner in the pool can't decide independently. In other words, the number of shares a miner find is limited to other requirements. The approximation should be improved.

Figure 2 displays a short-time-reward-oriented miner's choice in a pool with 40% hash power of the network. It shows the value




(a) 10 percent hashpower of the pool




(b) 20 percent hashpower of the pool

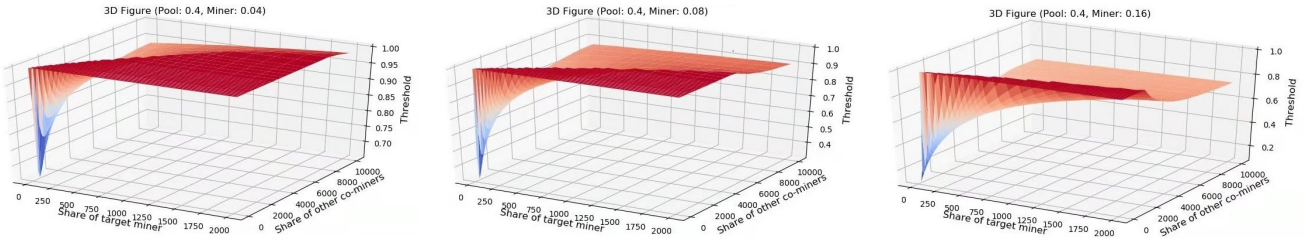
**Figure 2: The threshold of s to incentivize a miner with power proportion of 0.1 and 0.2 in the pool.**

that $s$ needs to be lower than to incentive a miner with 10 percent in Figure 2a and 20 percent hash power of the pool in Figure 2b who has found different numbers of shares as the number of shares the whole miner changes, in which the x-coordinate represents the number of shares the whole pool has found, and the y-coordinate represents the value $s$ needs to be lower than to incentivize the miner to submit the block immediately. Long-time-reward-oriented miners are more likely to withhold blocks. Figure 3a gives a 3D display of the threshold value of $s$ when the number of shares found by a long-time-reward-oriented miner and the whole pool change, and Figure 3b is a heat map of that when the miner's power proportion is 0.1, 0.2 and 0.4.

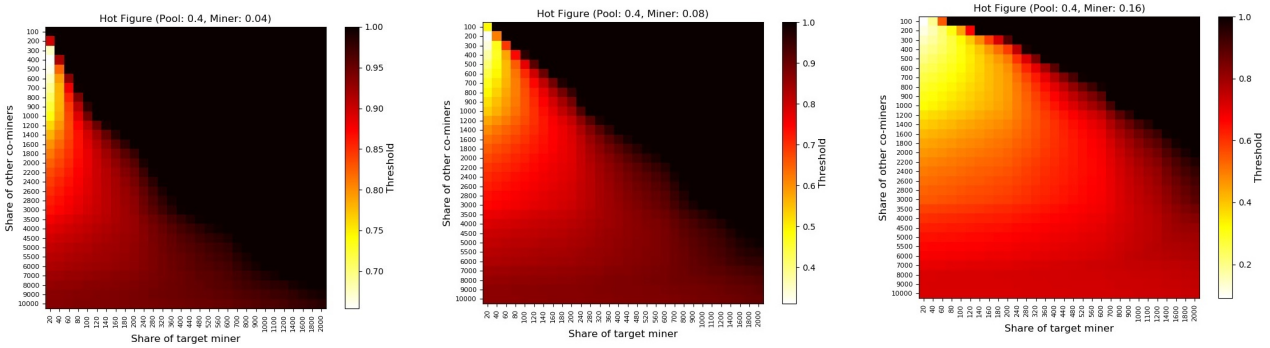
### 6.2 Selfish Mining When Profitable

Short-time-oriented managers wouldn't launch a selfish mining attack, as Theorem 2 expresses. Our simulation results verify this verdict— the manager doesn't launch an attack as $c$,$s$, and the mining power of the manager and the pool change. Thus, the manager mentioned in the following paragraphs in this subsection is long-term-reward-oriented.

Figure 4 gives a 3d-display of reward expectation of publishing the block immediately and a lower bound for the reward expectation of selfish mining alone when $s$ and $c$ change in some special cases for instance. The mining power of the mining pool is 0.4, and that of the manager is 0.15,0.2,0.25 respectively, while the numbers of shares submitted by the manager and the whole pool are 1000/5000 and 7000/11000, respectively. Figure 5 fixes $c$ to 0.5 and shows a




(a) 3D display




(b) Heat map

**Figure 3: The threshold of s to incentivize a miner with power proportion of 0.1 and 0.2 in the pool.**

lower bound for the excess reward expectation of selfish mining to honest behaviour in different cases to show the result more clearly.

We should emphasise Figure 4 and Figure 5 show a lower bound, which regards the reward when $sum2 = \sum b_i + D * \frac{\alpha_p - \alpha_m}{\alpha_p}$ to be $E_1$ instead of the real reward expectation. It can be seen that the curve of excess reward expectation of selfish mining goes up as $s$ rises on the left, but goes down on the right. An explanation for this is when the fraction of the number of shares submitted by the manager among all the shares is relatively low, the loss of allocated reward for shares is relatively little too, and it becomes less as $s$ decreases, reducing the opportunity cost of selfish mining, and it's the primary factor in this case. On the other hand, when the number of shares in total is relatively large, other miners' marginal allocated reward for continuing mining and submitting shares is little. It becomes less as $s$ decreases, and it's the primary factor in this case.

We implement a Monte Carlo simulator to simulate the manager's attack. We use expected relative extra rewards(RER) to show the performance of the attack. The RER of a strategy S of a mining entity x can be expressed as $RER_x^S = \frac{R_x^S - R_x^H}{R_x^H}$, in which $R_x^H$ denotes the reward expectation of honest mining.

Both Table 3 and Table 4 display the case when the reward allocation function is adjusted proportional function. Table 3 shows the RER of a manager with a hash power of 0.25 and two miners with a hash power of 0.05,0.1 in a pool with a hash power of 0.4 as $s$ and $c$ change when no miners launch withholding attack. To behave more like an honest manager, the manager might attack only when it finds a block before others itself, which is called strategy 'A' in Table 3. While strategy 'B' means the manager may withhold blocks in any case. The reward of the two strategies with different parameters is compared. A negative RER means getting less reward than the case when every one is honest. Table 4 shows RER of a manager in a pool as the hash power of the manager and the pool change, when $c = 0.5$, $s = 0.8$ and no miners launch withholding attack. Table 5 shows RER of a manager who may withhold block any time when the reward allocation function is IC function. To simplify the simulation process, the selfish mining continues only until anyone finds another block. For instance, a miner in the pool submits a block, and the manager chooses to withhold it and selfish mines following it, and if another miner in the pool finds a block before any others in the network, the manager stops selfish mining and propagates the first block. This is also in case that the miners detect the manager's attack and their loss themselves.

It is proved selfish mining requires a hash power of 0.25 to be profitable when $c = 0.5$ [11], while Table 3 shows the manager with 0.2 of hash power in the whole network when $c = 0.5$ gains extra rewards successfully no matter whether it restricts its attacking case. Thus, the attack we propose requires less hash power than selfish mining to be profitable. RER of the manager when applying strategy 'B' is higher than 'A', while that of miners is lower in Table 3, which means the manager can get more extra reward if possible to withhold block in any case, decreasing the rewards of miners in the pool. Table 3 also shows that the extra reward expectation of the manager rises as $s$ decreases. And when $c$ is larger than 0.7

**Table 3: The RER(%) of a manager with a hash power of 0.25 and RER(%) of two miners with a hash power of 0.05/0.1. 'A' means only when the manager finds a block before others do will it attack, while 'B' means the manager may withhold block at any case. 'Ma' denotes the manager, while 'Mi' denotes the miners: Adjusted Proportional Reward Function.**

| $s$ | $Ma/Mi$ | $c = 0.4$ | | $c = 0.5$ | | $c = 0.6$ | | $c = 0.7$ | | $c = 0.8$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $A$ | $B$ | $A$ | $B$ | $A$ | $B$ | $A$ | $B$ | $A$ | $B$ |
| 0.9 | $Ma$ | 0.23 | 0.36 | 0.65 | 0.84 | 1.10 | 1.53 | 2.22 | 2.91 | 3.18 | 4.27 |
| 0.9 | $Mi$ | −1.35, −1.39 | −2.12, −2.09 | −1.98, −2.01 | −2.83, −2.87 | −1.74, −1.80 | −2.54, −2.61 | −0.34, −0.40 | −0.50, −0.58 | 1.51, 1.50 | 0.73, 0.75 |
| 0.8 | $Ma$ | 0.28 | 0.41 | 0.74 | 0.95 | 1.72 | 2.31 | 2.56 | 3.46 | 3.94 | 5.03 |
| 0.8 | $Mi$ | −2.53, −2.51 | −3.35, −3.38 | −2.90, −2.97 | −3.87, −3.82 | −2.21, −2.20 | −2.89, −2.85 | −0.01, −0.01 | −0.02, −0.02 | 1.48, 1.50 | 0.85, 0.86 |
| 0.7 | $Ma$ | 0.83 | 1.12 | 1.42 | 1.87 | 2.18 | 2.94 | 3.29 | 4.36 | 4.58 | 5.91 |
| 0.7 | $Mi$ | −3.75, −3.85 | −4.93, −4.96 | −2.96, −2.97 | −3.71, −3.84 | −3.24, −3.22 | −4.21, −4.25 | −1.20, −1.32 | −1.64, −1.68 | 1.02, 0.96 | 0.43, 0.39 |
| 0.6 | $Ma$ | 1.12 | 1.64 | 1.87 | 2.55 | 2.84 | 3.71 | 4.05 | 5.36 | 5.51 | 7.19 |
| 0.6 | $Mi$ | −5.54, −5.76 | −7.24, −7.25 | −5.70, −5.48 | −7.31, −7.23 | −4.37, −4.50 | −5.83, −5.91 | −2.24, −2.15 | −2.91, −2.78 | 1.02, 0.96 | 0.43, 0.39 |
| 0.5 | $Ma$ | 2.23 | 2.89 | 2.75 | 3.68 | 4.04 | 5.32 | 5.19 | 6.74 | 6.75 | 8.95 |
| 0.5 | $Mi$ | −8.76, −8.52 | −11.23, −11.09 | −7.52, −7.36 | −9.94, −9.98 | −5.85, −5.97 | −7.71, −7.63 | −3.45, −3.39 | −4.58, −4.43 | −0.31, −0.27 | −1.32, −1.34 |

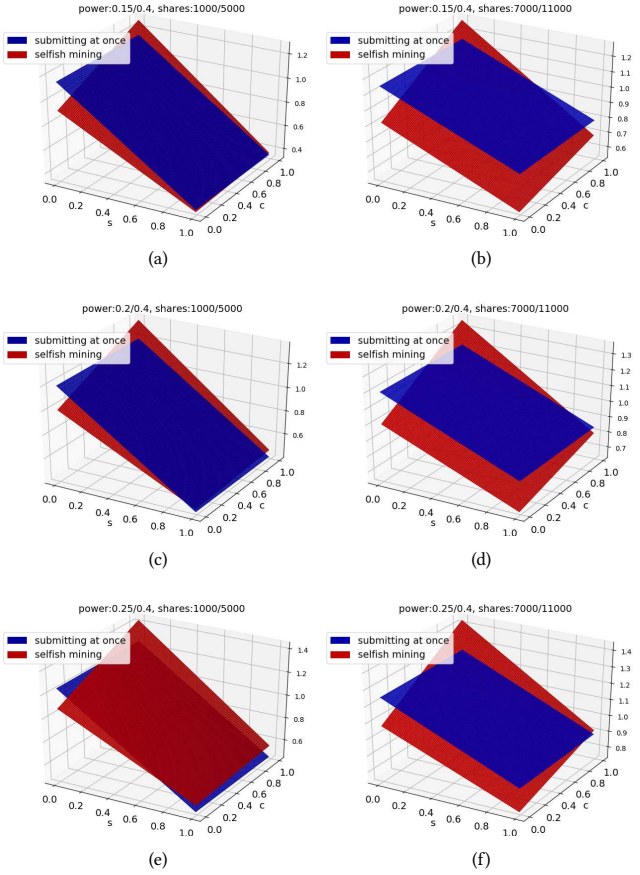
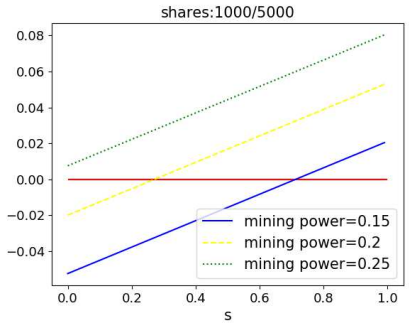
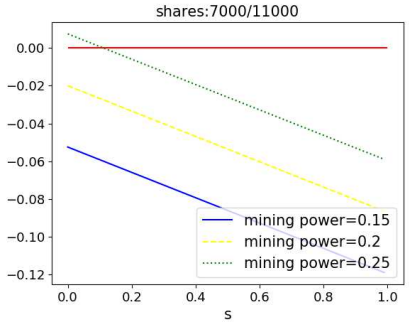



**Figure 4: Rewards of propagating the block at once and a lower bound for the rewards of selfish mining in different cases.**




(a) Shares:1000/5000




(b) Shares:7000/11000

**Figure 5: The threshold of s to incentivize a miner with power proportion of 0.1 and 0.2 in the pool.**

**Table 4: The RER(%) of a manager in a pool when $c = 0.5$, $s = 0.8$: Adjusted Proportional Reward Function.**

| $\alpha_p$ | $\alpha_m = 0.2$ | $\alpha_m = 0.225$ | $\alpha_m = 0.25$ |
|---|---|---|---|
| 0.3 | 0.03 | −0.11 | 0.08 |
| 0.35 | 0.46 | 0.01 | 0.19 |
| 0.4 | 0.86 | 1.02 | 0.74 |

in Table 3, the extra reward of the miner could compensate for the loss of other miners, which implies the manager could give some of its extra rewards to miners to avoid rational miners' hopping out. When $c$ is larger than 0.8 in Table 3, the miners attacked who behave honestly actually gain extra rewards themselves, which

**Table 5: The RER(%) of a manager with a hash power of 0.25 and RER(%) of two miners with a hash power of 0.05/0.1. The manager may withhold block at any cases. 'Ma' denotes the manager, while 'Mi1' and 'Mi2' denote the miners with power 0.05 and 0.1 respectively: IC Reward Function.**

| | $c = 0.4$ | $c = 0.5$ | $c = 0.6$ | $c = 0.7$ | $c = 0.8$ |
|---|---|---|---|---|---|
| $Ma$ | 0.67 | 1.13 | 2.08 | 3.52 | 4.94 |
| $Mi1$ | −2.45 | −2.56 | −1.74 | −0.21 | 1.08 |
| $Mi12$ | −2.41 | −2.45 | −1.77 | −0.19 | 1.02 |

we explain as a substitute for cooperating selfish mining without conspiring based on trust. Since the reward expectation for a valid block when the reward function is IC function is less than that when the reward function is adjusted proportional function and s is low, RER in Table 5 is lower.

Another interesting result is that in Table 3, when c rises from 0.4 to 0.5 and s is relatively close to 1, the rewards the miners in the pool get decrease, while when s is relatively low, the rewards the miners get increase. An explanation for this may be: the increasing of c has two consequences: the manager more likely to attack and the possibility of their becoming the main branch becoming higher. When c is relatively low, and s is relatively close to 1, the first is the main influencing factor, and the manager attacking tends to plunder miners in the pools' reward. When c is relatively high, the second is the main influencing factor, which may be beneficial for the miners. The reduction of the miners' loss when s is low is hard to explain intuitively.

Three points should be emphasized. First, the selfish mining only continues until anyone finds another block in our simulation, but in fact, the manager could continue selfish mining if only caring about its extra rewards, not considering the attack detected by miners. Thus, the RER of both the attack we propose and selfish mining could be higher.

Second, the RER of the selfish mining strategy rises as $s$ decreases, and a low value of $s$ ensures short-term-reward-oriented and long-term-reward-oriented miners in the pool wouldn't withhold their block in most cases. However, a low value of $c$ also increases honest miner's loss; therefore, $c$ needs to be high to make extra reward large enough to compensate for their loss, or the attack is only able to last for a short period. We also stimulate an attacker launching a selfish-mining attack with a hash power of 0.25 and the $c$ is 0.8. The RER comes out to be 11.25%, but it's 6.53% if the attacker publishes the private chain at once when finding the next block. When the reward function is IC function and c is high, our attack may not be profitable compared to traditional selfish mining alone. But when setting the reward function to be adjusted proportion function, the manager's reward expectation rises and the manager in our attack keeping selfish mining instead of publishing at once would make our attack more profitable than selfish mining.

Third, the attack would be easy to detect, if the manager simply launches the attack, the manager could act in a rational way to increase the difficulty for miners to detect it. In case the miners hop out, dynamically maintaining the balance between RERs of miners and the manager is needed. The manager can decrease

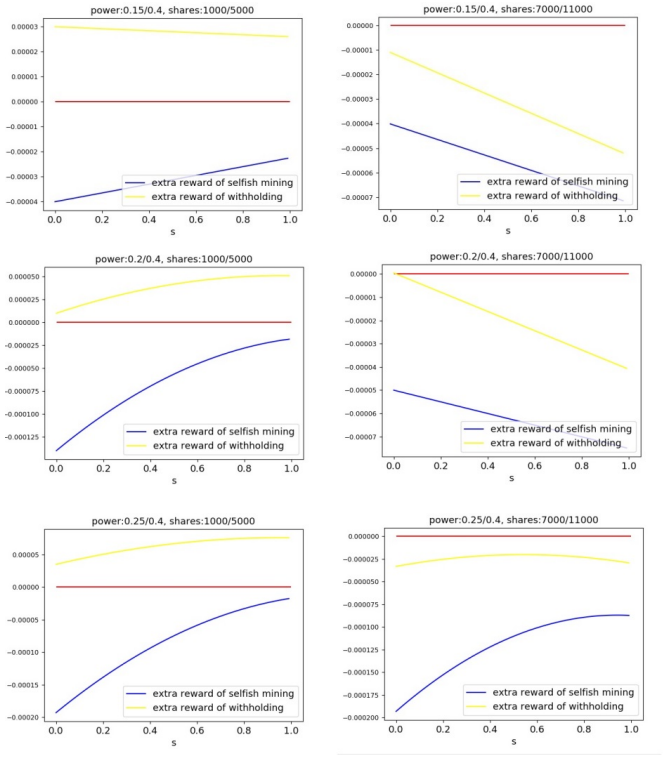



**Figure 6: The extra reward expectation of withholding and selfish mining during a share's interval when c=0.5**

the frequency of attacking, choosing to act honestly with some possibility even if attacking is more beneficial, for instance. We discuss this further in Section 7.

## 6.3 Improve the Effect by Combining with Withholding Attack

As Theorem 1 and Theorem 2 in Section 5.2 expresses, in our simulation, short-term-reward-oriented managers wouldn't launch an attack. The manager mentioned in the following paragraphs in this subsection is long-term-reward-oriented.

Figure 6 fixes $c$ to 0.5 and shows the excess reward expectation of withholding and selfish mining for a share's time. It should be emphasized that the reward of withholding lower than those of the other two in Figure 8 doesn't mean selfish mining is not profitable, because selfish mining for a longer time if miners out of pool haven't found blocks may lead to higher reward.

The simulation process would be of too high cost if allowing the manager to choose whether to continue withholding after every share; thus in our simulation, if choosing to withhold, the manager would keep the fraction of mining power which continues submitting shares until any miners in the network find another block. Table 6 shows the RER of a manager with a hash power of 0.25 and two miners with a hash power of 0.05,0.1 in a pool with hash power of 0.4 as $s$ and $c$ change when no other miners launch withholding attack.

**Table 6: The RER(%) of a manager with a hash power of 0.25 and RER(%) of two miners with a hash power of 0.05/0.1. 'Ma' means the manager and 'Mi' means the miners**

| $s$ | | $c = 0.4$ | $c = 0.5$ | $c = 0.6$ |
|---|---|---|---|---|
| 0.9 | $Ma$ | 0.28 | 0.74 | 1.22 |
| 0.9 | $Mi$ | −1.42, −1.43 | −2.04, −2.05 | −1.88, −1.89 |
| 0.8 | $Ma$ | 0.33 | 0.56 | 1.84 |
| 0.8 | $Mi$ | −2.56, −2.54 | −2.96, −3.01 | −2.28, −2.27 |
| 0.7 | $Ma$ | 0.93 | 1.54 | 2.31 |
| 0.7 | $Mi$ | −3.84, −3.96 | −2.96, −3.01 | −3.35, −3.34 |
| 0.6 | $Ma$ | 1.39 | 2.01 | 2.97 |
| 0.6 | $Mi$ | −5.66, −5.78 | −5.81, −5.63 | −4.52, −4.61 |

The RER of the manager can be increased if allowing the manager to make choice of withholding after every share and keep selfish mining for longer periods. And the RER of the manager is higher than joining a pool and launching a block withholding attack.

In conclusion, the attack may even profit the miners who act honestly in the pool if the network connection advantage is large. Thus, rational miners might not hop out of the pool even if they detect the attack.

## 7 DISCUSSION AND EXTENSION

Miners in the pool may detect the manager's attack when they submit block and don't see it propagated. If $c$ is large enough, the manager would like to convey the message to rational miners it has network connection advantages, and the rational miners may not hop out if they find the manager is attacking, but their reward expectations actually rise in a long time. When the network advantage is little, the manager acts honestly. And when the network advantage is big, the manager selfish mines, withholds or discards blocks with possibility at random instead of deterministically if reward expectation of attacking is larger. This decrease the possibility of miners detecting the attack and hopping out, at the expense of larger extra reward. What makes the attack more practical, the attack decreases the reward expectation of miners out of the pool, making the reward of mining in the pool more likely to be higher than mining out of it. Thus, it's easier to keep and attract rational miners.

The incentivizing measures may not succeed when a large pool launches a BWH attack on our victim pool, allowing them to change power splitting. However, the optimal withholding strategy for the attacker is to attack as many pools as possible, and the optimal power splitting strategy only split a fraction of the power to do infiltrate mining in the pool. The withholding decreases our attack's extra rewards comparing to all miners being honest, but the manager's reward would be lower without adjusting the reward allocation function and conduct our attack. And if $c$ is large enough, the manager and miners in the pool can still gain from the attack we propose.

The miners in the pool may conduct improvement of BWH attacks, making the situation more complicated. For instance, if the miners in the pool launch FAW attack instead of block withholding attack, the attacker's submitting block instead of discarding it is actually benefitting the pool. Thus, though FAW attack is hard to avoid for the manager, its negative effects on profits of our attacks might be less than that of attackers with large power's withholding attack.

## 8 CONCLUSION

We study rational pool manager's strategy to incentivize miners in the pool not to withhold blocks and gain extra rewards. Miners are modeled into short-time-reward-oriented and long-time-reward-oriented ones, which is more realistic, reasonable, and comprehensive assumptions about miners' decision comparing to past researches. It is verified that the manager can incentivize miners with not too large hash power and gain extra rewards at the same time with enough hash power and network connection advantages. The hash power and network connection advantages it requires for the attack to be profitable are lower than that of selfish mining attack, and the reward expectation is higher. The manager may apply this strategy of whether to withhold and discard other miners' blocks in more advanced and comprehensive attacks such as stubborn mining to get more extra rewards. We discuss some improvements to the attack we propose.

To conclude, our paper is a beginning, and the question as well as the analysis approach, might be extended to more general situations; for instance, multiple players cooperate on selfish mining to increase the rewards. It may lead to more practical concerns for the stability of the Bitcoin protocol in this way. More work is needed and welcomed.

## 9 ACKNOWLEDGMENTS

This work is supported by Chinese National Research Fund (NSFC) No.61702330.

## REFERENCES

[1] Joseph Boneaau and Andrew Miller. 2015. Research Perspectives and Challenges for Bitcoin and Cryptocurrencies. *Security&Privacy* (2015).
[2] Arthur Gervais et al. 2016. On the Security and Performance of Proof-of-Work Blockchains. *ACM Conference on Computer and Communications Security* (2016).
[3] Ayelet Sapirshiten et al. 2016. Optimal Selfish Mining Strategies in Bitcoin. *Financial Cryptography* (2016).
[4] Christian Decker et al. 2013. Information Propagation in the Bitcoin Network. *International Conference on Peer-to-peer Computing, IEEE* (2013).
[5] Ethan Heilman et al. 2015. Eclipse Attacks on Bitcoin's Peer-to-Peer Network. *Proc. of the USENIX Security Symposium (Security)* (2015).
[6] Loi Luu et al. 2015. On Power Splitting Games in Distributed Computation: The Case of Bitcoin Pooled Mining. *Computer Security Foundations Symposium, IEEE* (2015).
[7] Okke Schrijvers et al. 2016. Incentive Compatibility of Bitcoin Mining Pool Reward Functions. *Financial Cryptography* (2016).
[8] Shang Gao et al. 2019. Power Adjusting and Bribery Racing: Novel Mining Attacks in the Bitcoin System. *ACM Conference on Computer and Communications Security* (2019).
[9] Yujin Kwon et al. 2017. Be Selfish and Avoid Dilemmas: Fork After Withholding (FAW) Attacks on Bitcoin. *ACM Conference on Computer and Communications Security* (2017).
[10] Ittay Eyal. 2015. The Miner's Dilemma. *Security&Privacy* (2015).
[11] I.Eyal and E.G.Sirer et al. 2014. Majority is not Enough: Bitcoin Mining is Vulnerable. *Financial Cryptography* (2014).
[12] Hanqing Liu, Na Ruan, Rongtian Du, and Weijia Jia. 2018. On the Strategy and Behavior of Bitcoin Mining with N-attackers. *ACM Asia Conference on Computer and Communications Security* (2018).
[13] Kartik Nayak and Srijan Kumar et al. 2016. Stubborn Mining: Generalizing Selfish Mining and Combining with an Eclipse Attack. *European Security&Privacy* (2016).